DOCUMENT RESUME

ED 251 130 JC 840 570

AUTHOR Flamm, Ann L.
TITLE The Reading Area Community College Faculty Development Project Final Report: Classification and Integration of Software/Computer Literacy.
INSTITUTION Reading Area Community Coll., Pa.
PUB DATE 30 Sep 84
NOTE 68p.; Developed with support from a Strengthening Program Grant.
PUB TYPE Reports - Descriptive (141)

EDRS PRICE MF01/PC03 Plus Postage.
DESCRIPTORS Community Colleges; *Computer Literacy; *Computer Science Education; Course Descriptions; *Courseware; *Mathematics Instruction; Program Descriptions; Two Year Colleges; Videotape Recordings
IDENTIFIERS *Software Evaluation

ABSTRACT

Results are presented from a faculty development project conducted at Reading Area Community College (RACC) to examine and organize the college's software holdings; prescribe support software for mathematics courses; create a computer literacy unit for the Foundations of Mathematics course; and view and describe RACC's mathematics videotapes. The report includes a review of the rationale for the project, a description of the methodology used, a discussion of time frames, a narrative of project outcomes, and a presentation of conclusions and recommendations. The bulk of the report consists of appendices, which provide: (1) software recommendations, which include information on the recommended package and its vendor and provide a brief description of contents; (2) a proposal for microcomputer use at RACC in academic and support areas; (3) an inventory of the software and math videotapes available at RACC, which identifies and describes the software packages and videotapes and offers recommendations for their use; (4) course outlines for RACC's Foundations of Mathematics and Developmental Mathematics courses; and (5) the computer literacy unit, which includes information on the history of the computer, an overview of programming languages, and an introduction to basic programming. (LAL)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
G. Wilkins
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
☐ This document has been reproduced as received from the person or organization originating it.
☒ Minor changes have been made to improve reproduction quality.
• Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

ED251130

THE READING AREA COMMUNITY COLLEGE

FACULTY DEVELOPMENT PROJECT FINAL REPORT

Classification and Integration

of Software/Computer Literacy

Conducted

by

Ann L. Flamm
Assistant Professor
Division of Science/Mathematics

Submitted for

Use of the College Community

on

September 30, 1984

Staff assistance and financial support for this project was provided through a Strengthening Program Grant (Title III of The Higher Education Act of 1965).

JC 840 570

Reading Area Community College
P. O. Box 1706
Reading, Pennsylvania 19607
215-372-4721

A. Review of Project Rationale

During the last two years the educational focus has been on computers and science/math instruction; this project focused on uniting these two areas. It was a four-part endeavor to enhance the mathematics courses by using the available software and hardware in a more efficient manner.

Supplemental instructional materials such as software are housed in the Learning Resources Center (LRC) and more recently in the Science/Math Lab (Room 419). The list of software was incomplete and no descriptive material existed. If a student expressed a desire to use the software, a recommendation was made by the instructor. Usually the selection of the software was done by one of the clerks in the LRC according to the topic desired by the student. The follow-up on software usage was weak. There is a definite need for a complete list of software, descriptive material, and a controlled method of software usage.

The Foundations of Mathematics course has been expanded to include chapters on Algebra and Statistics. At recent conferences, computer literacy was also included among the topics taught in the Foundations course. This was also one of the recommendations of the Basic Skills Articulation Team. Since the Foundations course is a transfer course, the need for the students to be exposed to computers and the BASIC Language seemed evident in light of the current technology and the increasing use of computers in all areas of education. The Foundations course is also a prerequisite for the Statistics course, and an introduction to the computer in the Foundations course would allow the Statistics instructor to use the

computer as a calculator and the statistical software packages as instructional material.

The mathematics videotapes have been housed in the Learning Resources Center for four years. There was a need to re-examine and document the content of each of the twenty (20) videotapes to improve student use of these tapes.

Therefore, a four-part faculty development project was proposed to address the areas previously mentioned. The goals of the proposal were as follows:

1. Examination and Organization of Software Holdings
   a. Examine the current software to see which disks were copyrighted.
   b. Classify the software as drill and practice or tutorial with or without a management feature.
   c. Assign the software to specific topics within the mathematics courses and indicate the appropriate mathematics course that would use the software.
2. Prescription of Support Software for the Mathematics Courses
   a. Develop the formation of a prescription sheet with a complete listing of software organized according to the mathematical topics covered in the DSP Mathematics class. (After an individual diagnosis of each student, the appropriate software would be selected on the prescription sheet.)
   b. Adjust the DSP and Foundations of Mathematics course syllabi to effect the above changes.

3. Creation of a Computer Literacy Unit in the Foundations of Mathematics Course

   a. Gather material for student orientation to the computer.

   b Organize information to instruct students in basic programming and the use of the computer as a calculator which can be used in other course units.

4. View and Document Descriptions of the Mathematics Videotapes

By completing the project, the supplemental instructional material for the mathematics courses would be organized in a more useful manner for students and instructors. There also would be greater instructor awareness of computers as instructional aides, and students would gain working knowledge of computers and the BASIC Language.

B. Project Methodology

The order in which project objectives were addressed was reversed according to the availability of materials and personnel.

1. The twenty (20) videotapes were viewed in the Nursing Library. The content on all but two (2) of the tapes was good to excellent. The quality of the videotapes ranged from fair to poor. A description of the material discussed was noted for each videotape with a recommendation on how to use the tape and with what course it would be most beneficial. A recommendation was made that two videotapes be deleted only after they are viewed by other science/math personnel. (See Appendix D.)

2. The collection of information for the Computer Literacy Unit became the most interesting and time-consuming activity. The "small" unit has been revised three times and is eighteen (18)

pages in length. The unit contains three main content areas: the history, the computer, and beginning programming information.

The Computer Literacy Unit is designed for use in the Foundations course; therefore, all the programs and examples are mathematical in nature. The unit is broad enough in scope that with minor revisions any part of the Computer Literacy Unit could be adapted to other courses. (See Appendix F.)

3. The evaluation of software holdings and the prescription sheet of support software for the mathematics courses were dependent on the same objectives. The current software was examined, classified, and assigned to the appropriate categories. (See Appendix C.) The information was incorporated in both the Foundations of Mathematics and the DSP Mathematics course syllabi. (See Appendix E.) After achieving the above objectives the software was ready to be used in a menu-driven program designed for the Corvus. At this point the objectives had to be adjusted. Due to the lack of monies, RACC Computer Lab Assistant Brian Schneider's microcomputer implementation project was not funded. (See Appendix B.) Therefore, the completion of a formal prescription sheet was amended and the project goals were adjusted accordingly.

The organization of the material on the Corvus became an obsession. At a "Basic Skills and Microcomputer" conference in New York, an instructor from Loyola University, Louisiana, presented a menu-driven program designed for the Corvus. After many unsuccessful attempts, contact was made with the university and a coordination of activities followed.

After explaining to Brian Schneider the dilemma, he graciously agreed to speak with Loyola's programmer and obtain information necessary to develop on a smaller scale a menu-driven program for the mathematical material on the Corvus.

Roger Kimpland, RACC Coordinator of Audio-Visual Services, also agreed to help re-arrange the mathematical material on the Corvus. The mathematical information is scattered throughout six to eight volumes that are not in numerical sequence.

After examining the material on the Corvus, the following objectives are being coordinated to complete the original goals in the project:

a. Roger Kimpland and Ann Flamm are working to identify the mathematical material that was placed on the Corvus, and Roger will then select the volumes for mathematical use.

b. Roger Kimpland and Brian Schneider are also testing recently purchased mathematical software for possible placement on the Corvus.

c. Brian Schneider will complete the menu-driven program that accesses the mathematic volumes on the Corvus after all the material is gathered and sequenced.

d. The current listing of software in the appendix will be adjusted accordingly, and a formal prescription sheet to coordinate the Corvus material with the mathematical software will be constructed.

e. The management features of selected disks will be organized for the instructor's use.

C. Project Objectives (See Section B above.)

D. Project Duration

The project began on June 25, 1984, and the main objectives were completed by September 17, 1984. The original goal of coordinating projects with Brian Schneider and the adjusted objectives (See B above.) will be completed by December, 1984. Significant time was spent gathering and revising the information for the Computer Literacy Unit. These additional efforts as coordinator were not anticipated, but were incorporated to achieve the goals of the project. The review of the software and videotapes schedule was followed.

E. Project Narrative

1. The examination and organization of the software holdings was incorporated into the syllabi of both the DSP and Foundations courses. A separate list was developed for instructional purposes. The Learning Resources Center and the instructor coordinated efforts to account for student usage of the software. There will no longer be recommendations, but specific disks required. Additional copies of the software material will be given to other instructors.

2. The final formation of a prescription sheet will be completed when the Corvus material is organized. This information will also be given to other instructors.

3. The Computer Literacy Unit will be used in the Fall Term, 1984, and revisions made as the term progresses. As noted before, the unit is broad-based and can be used in other courses in sections or as a whole.

4. The mathematics videotapes were incorporated into the Foundations syllabus, and a separate listing was arranged for instructional purposes.

F/G. Conclusions/Recommendations

1. The software purchased for instructional use should be previewed and then tested to see if the material can be placed on the Corvus. If it cannot, a hard disk which contains a larger package of mathematical topics may be purchased.

There is a broad range of topics covered by the software. Additional copies of the present software should be purchased or new disks purchased to enhance the topics presently covered by our software. There is a recommended list of software (or hard disk versions) attached to the report. (See Appendix A.)

Instructors should use a more formal approach in using the software for instructional purposes. The software is excellent as tutorial material and should be required as part of course instruction.

The Learning Resources Center should keep current listings of the software and adequate descriptions on each disk purchased. There should be better communication among departments to list all the software housed in the various labs.

The management aspect of several disks involve tedious procedures to use those particular features adequately. The selection of one disk with a management feature and the use thereof would assist the instructor in instructional time or preparation time after the initial procedures are established.

The final recommendation would be to reward Brian Schneider and Roger Kimpland with a monetary payment for their assistance in achieving the goals and objectives of this project and expanding several of the objectives.

2. When the prescription sheet is in final form, other instructors may derive benefits from the listing or format.

3. It is recommended that the Computer Literacy Unit be used in all Foundations of Mathematics classes. A copy of the unit should be given to each instructor when it is in final revised form.

4. All mathematics instructors should make an effort to use the videotapes as tutorial or supplemental material until adequate time is found to create their own videotapes.

Appendix A:

Software Recommendations

PACKAGE	VENDOR
ALGEBRA DRILL II	CONDUIT SIMPLIFYING EXPRESSIONS, WORD PROBLEMS INVOLVING DISTANCE, RECTANGLES, & MISSING SOLUTIONS.
BASIC MATH SKILLS	EDUCATIONAL ACTIVITIES +−*/ WHOLE NUMBERS & FRACTIONS TABLES, ROUNDING, DECIMALS, GRAPHS, MEAN, MEDIAN, MODE.
BROKEN ZILBOWS	PROGRESSIVE SOFTWARE ADDITION, SUBTRACTION, MULTIPLICATION, AND DIVISION OF FRACTIONS.
COMPU-MATH DECIMALS	EDU-WARE SERVICES, INC. PRE-TEST, CONVERSION TO DEC, ROUNDING DEC, PERCENT, +−*/ OF DECIMALS.
COMPU-MATH FRACTIONS	EDU-WARE SERVICES, INC. DEFINITIONS OF PARTS, PRE-TEST DENOMINATORS, +−*/ OF FRACTIONS.
EDU-WARE ALGEBRA	EDU-WARE SERVICES, INC. NUMBER LINE OPERATIONS, SETS, EVALUATING EXPRESSIONS, RULES FOR EQUATION REDUCTION.
INDIVIDUAL STUDY CENTER	TYC SOFTWARE SIMPLE ADDITION, SUBTRACTION, MULTIPLICATION, AND DIVISION.

MAGIC GRID	IDEAL SCHOOL SUPPLY CO. DRILL IN SOLVING FOR ONE UNKNOWN VARIABLE. CONTAINS TWO GAMES AS REWARD.
MATH PACKAGE I	MICRO LEARNINGWARE MATH DRILL, SPEED DRILL, PLACE VALUE, STRINGS.
MATH PACKAGE II	MICRO LEARNINGWARE BAGELS, STONES, FACTOR, DECIMALS, FRACTIONS.
NUMBER SERIES	PROGRAM DESIGN, INC. 8 LESSONS, QUIZ IN FINDING THE NEXT TWO NUMBERS IN A GIVEN SERIES.
SOLVING WORD PROBLEMS I	N & S LEARNING STRATEGIES 40 SHORT DRILLS IN SOLVING WORD PROBLEMS INVOLVING ONE OPERATION.
SOLVING WORD PROBLEMS II	N & S LEARNING STRATEGIES 40 SHORT DRILLS IN SOLVING WORD PROBLEMS INVOLVING TWO OPERATIONS.
STATISTICS	EDU-WARE SERVICES, INC. MEAN, VAR, ST. DEV, CHI-SQ, T-TEST, CORRELATION, NORMAL DISTRIBUTIONS, AND REPORTS.

ALGEBRA DRILL I	CONDUIT ALGEBRAIC AND NUMBERIC FRACTS, SLOPES & EQUATIONS OF LINES, PERCENT, AND SIGNED NUMBERS.
COLLEGE BOARDS PART 2	KRELL MATHEMATICS DRILL.

Appendix B:

Microcomputer Implementation

Proposed Project

MICROCOMPUTER IMPLEMENTATION

A. Proposed Objectives

The objective is to allow persons from academic and non-academic areas to use the software on the Corvus in a user-friendly and efficient manner.

B. Scope and Contents

The scope of the project is to catalog and organize the software available on the Corvus hard disk. This includes both Basic and Pascal usage. The project phases are: analysis, development, and implementation. Documentation, a fourth phase, will be included in the development phase as on-line help menus.

C. Benefits

The proposed implementation is to benefit both the academic and support areas. The benefits will derive from being able to determine the applicability, availability, location, and use of software on the Corvus.

D. Duration

Analysis phase	2 weeks	(8 working days)	X	13.20	=	$ 844.80
Development phase	6 weeks	(24 working days)	X	13.20	=	2,534.40
Implementation phase	2 weeks	(8 working days)	X	13.20	=	844.80

E. Project Materials

All materials are currently available either in the LRC or Room 409.

F. Time

All costs are directly due to time.

Brian Schneider

Appendix C:

Software Inventory

Software	Corvus	Description	Recommendation
Algebra I - Eduware Algebra II - Eduware		Menus are intricate. Student easily frustrated unless comfortable with computer. Ability to correct answers on test. Algebra I menu: definitions, number, line operations, sets, evaluating expressions, rules for equation reduction, post-test. Algebra II menu: addition of real numbers, multiplication of real numbers, solving equations, solving inequalities, post-test.	Use with a tutor in DSP, Elementary Algebra, Intermediate Algebra, or College Algebra. Instructional or tutorial depending on course level and student needs.
Algebraic Expressions		Has two levels. Step-by-step display. Uses negative integers.	DSP as a tutorial for Chapters 5, 6, and 7. Elementary Algebra and Foundations courses as review material.
Decimals - Eduware	Vol. 4	Menu is intricate. Cannot correct answers. Pre-test, conversion of decimals, addition, subtraction, rounding, multiplication, division, percent. Student easily frustrated unless comfortable with computer and content material.	CEC, DSP - use with a tutor.
Division Skills - Milton Bradley		Teacher-manager feature. Generates division problems with varying numbers of digits as divisors and dividends. Worksheets.	CEC - use in DSP as remedial.
Elementary Mathematics	Vol. 1	Only 3 sections contain material pertaining to mathematical concepts taught in courses. Learning to Round Off Numbers; Developing Speed in Math Drill - addition, subtraction, multiplication, division; Studying Factors of Numbers - Primes, Composites, Factors (difficult). Worksheets.	CEC, DSP, Elementary Algebra - use with a tutor.

Software	Corvus	Description	Recommendation
Factoring Machine		Has teacher-manager feature. Shows menu. Factors quadratic expressions. Able to correct errors. Menu: Distributive Property, Factor Games, Scheduled Quiz, Factor Machine (student can enter own problem and the computer will do the factoring).	Elementary Algebra, Intermediate Algebra
Fractions - Eduware	Vol. 3	Intricate menu. Student easily frustrated. Does not use lowest common denominator but reduces fractions to lowest term as the last step. Pre-test, definition of parts, denominators, addition, subtraction, multiplication, division, mixed numbers to improper fractions. Excellent instructions if student is comfortable with computer and content material.	CEC, DSP
Fractions - Mathware Ratio and Percents - Mathware		Teacher-manager feature. From extensive list of topics teacher can program disk - level of difficulty included. Teacher can review student's lesson. Excellent quantity and quality material. Verbal problems. Step-by-step instruction if requested.	CEC, DSP
Integers/Equations I & II		Teacher-manager feature. All operations with integers. Linear equations from one-step to multiple-step problems. Disk I contains integers (5 levels) and variables (4 levels). Disk II contains 15 levels of equations.	DSP, Elementary Algebra, Intermediate Algebra
Junior High Math Disk - Microphys	Vol. 28	Randomly generated problems. Teacher-manager feature. Possible quizzes. Mainly drill and practice. Step-by-step instruction. Menu: PJ802 Multiplication, PJ803 Division	CEC, DSP, Foundations

Software	Corvus	Description	Recommendation
		PJ804 Modular Arithmetic, PJ805 Proportion Problems, PJ806 Percent Problems, PJ807 Addition of Fractions, PJ808 Subtraction of Fractions, PJ809 Multiplication of Fractions, PJ810 Division of Fractions, PJ813 Decimals (Addition, Subtraction), PJ814 Decimals (Multiplication, Division)	
Lessons in Algebra		Menu 1, 2: Sets - Basic Vocabulary and Review of Basic Vocabulary. Menu 6, 7, 8: Exponents, Basic Vocabulary, Multiplication Like Bases, Division Like Bases. Menu 3, 4, 5: Axioms or Postulates. All have brief descriptions on topics covered, followed by examples, then questions, not step-by-step instruction. More of an overview.	Foundations, DSP, Elementary Algebra, Intermediate Algebra
Math Ideas		We received written programs associated with material in Mathematical Ideas, the Foundations of Mathematics text. Most of the programs are incomplete or do not run properly.	Delete
Order of Operations		Computational and procedural errors given after each problem. Able to correct errors. Step-by-step practice in simplifying numerical expressions containing the four basic operations and parentheses using the rules for order of operations.	DSP - use with a tutor. Elementary Algebra
Plane Facts About Geometry: Formulas, Shapes, and Skills		Handouts. Menu: Geometric Shapes, Perimeters, Rectangle and Square Areas, Parallelogram Areas, Trapezoid and Triangle Areas. Drill and practice. Graphic illustrations.	Foundations

Software	Corvus	Description	Recommendation
Understanding Lines and Triangles Geometry Basics		Handouts. Menu: Lines, Quiz on Lines, Triangles, Quiz on Triangles. Drill and practice. Graphic illustrations.	Foundations
Polynomials Practice Using Tiles: Discovering Algebra		Worksheets. Teacher-manager feature. Randomly generated problems. Menu: Introduction, Identify Collections, Addition, Subtraction, Multiplication, Division of Polynomials.	DSP, Elementary Algebra
Solving Equations and Inequalities: Discovering Algebra		Equations and inequalities with one variable. Randomly generated. Graphic presentation. Practical application. Worksheets.	DSP, Elementary Algebra
Microphys: Introductory Calculus - Set 7612		Randomly generated problems. Teacher-manager feature. Possible quizzes. Mainly drill and practice, no step-by-step instruction.	Calculus
Senior High Math Disk		Same description as above. PC703 Simultaneous Equations (2x2), PC704 Simultaneous Equations (3x3), PC706 Trig II, PC721 Linear Equations, PC724 Exponents and Logarithms, PC725 Verbal Problems - General, PC701 Quadratic Equations, PC739 Permutations I, PC740 Permutations II, PC741 Combinations, PC742 Permutations and Combinations, PC743 Probability I, PC747 Statistics I, PC748 Statistics II.	Elementary Algebra, Intermediate Algebra, College Algebra, Precalculus, Statistics, Foundations, Trigonometry

Appendix D:

Videotape Inventory

Although the viewing of the videotapes was time consuming, it was beneficial to re-discover the usefulness of the videotapes. The instructor recommends that the videotapes be used as supplemental instructional tools in the appropriate math courses. The use of the videotapes recommended for the algebra course will depend on the level of the course. They can be used as remedial or tutorial tapes. I would not recommend the videotapes be used as a substitute instructional method.

The videotapes recommended for the Foundations of Mathematics course will be used as supplemental instructional material with the assistance of the instructor or a tutor. These videotapes were incorporated in the course syllabus for Foundations of Mathematics.

The videotapes which are housed in the LRC have library user cards; therefore, the instructor is able to monitor the use of the videotapes by the individual students. The video viewing machine is operated by the student. The student therefore controls the videotape. One is able to stop the machine and take notes when necessary or rewind to review a portion of the tape.

There are 20 videotapes. The average length of the videotape is 35-40 minutes. The videotapes are black and white which tends to be a draw back with the color animation of today's videos. There are instances when the content is overshadowed by the poor quality of video taping.

Title	Description	Course Recommendation
Factoring Polynomials I	Covers distributive property adequately. Shows removal of common factors. Good relationship of distributive property - common factor in examples. Steps in factoring polynomials are excellent (step-by-step). Poor quality video but examples overshadow quality.	Elementary Algebra, Intermediate Algebra, College Algebra
Factoring by Grouping	Eight examples of factoring by grouping including cubes, squares. Sometimes poor video. Good tutorial tape.	Elementary Algebra, Intermediate Algebra, College Algebra
Factoring Polynomials II	Need Factoring by Grouping Video. General quadratic trinomial. Good examples with step-by-step instructions.	Elementary Algebra, Intermediate Algebra, College Algebra
Solving Quadratic Equations by Factoring	Explains ax^2+bx+c. Explains and demonstrates standard form. Good examples with the various forms ($x.x^2+7x=0$). Goes through steps of factoring from Polynomial I video. Completes with solving linear equations. Explains substitution process to see if the solutions are actually valid. Complete information.	Elementary Algebra, Intermediate Algebra, College Algebra
Solving Quadratic Equations by Formula	Explains and demonstrates completing the square. Develops quadratic formula by using completing the square and solving for x. Uses formula to solve quadratic equations. Several illustrations: stresses standard form, formula, imaginary roots - discriminant.	Elementary Algebra, Intermediate Algebra, College Algebra

Title	Description	Course Recommendation
Solving Linear Equations and Inequalities	Solves 4 problems: 2 inequalities, 2 equations. Doesn't tell you how to get common denominator. Multiplies by using distributive property. Sometimes close-ups of problems are blurry. Points out negative numbers in inequalities. Use with a tutor.	Elementary Algebra, Intermediate Algebra, College Algebra
Addition - Subtraction of Algebraic Fractions	Different method for finding the common denominator - uses prime factors (tree diagram) - otherwise good examples. Points out the difference between cancelling terms and cancelling factors. Points out common sign errors. Good tutorial tape.	Elementary Algebra, Intermediate Algebra, College Algebra
Multiplication and Division of Algebraic Fractions	Demonstrates side by side - fewer steps needed if factoring takes place first. Explanation of 1 in the numerator with several terms in the denominator. Repeats cancelling of factors. Good examples. Tutorial tape.	Elementary Algebra, Intermediate Algebra, College Algebra
Simplifying Complex Algebraic Expressions	Needs a worksheet. Use with a tutor. Solves 6 problems rapidly - complex fractions used, variables in denominator. Doesn't refer to numerator or use proper terminology. Method is OK - style and comments poor. Uses distributive property.	Elementary Algebra, Intermediate Algebra, College Algebra
Sets and Set Relations	Examples of set and set notation. Uses symbols. Material explained: null or empty set, set builder notation, subsets (proper, improper), equal sets. Good examples - clear and concise.	Foundations, Chapter 1

Title	Description	Course Recommendation
Set Operations and Properties	Questionable because content is not in current Foundations textbook. Initial material: sets, complements, intersection, union, De Morgan's Law - is useful. The properties of commutative, associative, distributive with relationship to sets is no longer a part of the Foundations text. Must have good understanding of algebra properties. Possible deletion of middle section. Use of overlays poorly done.	Foundations, Chapter 1; College Algebra
Cartesian Products and One-to-one Correspondence	Possible deletion. Introduces ordered pairs, Cartesian product of two sets, product of number of elements times number of elements - several illustrations. Lengthy discussion. One-to-one correspondence. Illustrations. Number of subsets possible use factorials.	Foundations, Chapter 1; College Algebra
Compound Propositions and Their Truth Tables	Illustrates the truth tables for and, or, if → then, biconditional, negation. Extensive explanation. Building process with examples. Explains or is used in the inclusive sense. Does biconditional with separate truth tables. Last example combines negation, or, biconditional, and.	Foundations, Chapter 2
Converse, Inverse, and Contrapositive	Follow-up after Compound Propositions. Good examples of the conditional and the converse, inverse, and contrapositive from the direct statement.	Foundations, Chapter 2
Venn Diagrams and Valid Arguments	Eight arguments presented. Good logical reasoning. Valid arguments are proven using truth tables in current text. Good supplemental information for motivated student.	Foundations, Chapter 2

Title	Description	Course Recommendation
Fundamental Principles of Counting and Permutations	Excellent presentation. Notation of Permutations - different than current text - student adjustment. Fundamental counting principle and permutations. Good instructional - tutorial. Repetitious with respect to basic principles.	Foundations, Chapter 10
Combinations	Notation of combination different than in current text. Comparison to permutations. Introduce formula - how derived. When tape gets to #269 possible deletion of material. Notes; combinations are collectors, sets, or bunches.	Foundations, Chapter 11

Appendix E:

Course Outlines

(Foundations and DSP Math)

Course Description
For
SM 112 Foundations of Math

3 Credit Hours

Text: Mathematical Ideas
Charles D. Miller
Vern E. Heeren

Ann Flamm, Assistant Professor
Reading Area Community College

This course is designed for students who plan to limit their study of mathematics to the minimum college requirement. This course places as much emphasis on the modern mathematical ideas and their meaning as on computation. The student will be introduced to computer literacy, set theory, logic, systems of numeration, finite mathematical systems, counting theory, probability, statistics, and additional topics in Euclidean geometry.

OVERALL COURSE OBJECTIVES

The purpose of the course, Foundations of Mathematics, is to give insight into some of the more uncommon areas of mathematical thought. As many of these areas require the learning of methods of investigation rather than memorization, the main goal is that the student should be able to transfer his knowledge of logical investigation of mathematics to other fields of study.

Upon completion of the course, the student should be able to:

I. Sets: define a set in three ways; define and use set relations; define, use, and illustrate with Venn diagrams complement, intersection, and union; recognize the following properties involving intersction and union: commutative, associative, distributive, and deMorgan's laws; define one-to-one correspondence, equivalent sets, finite sets, and infinite sets.

II. Logic: define or describe the nature of a mathematical system, its components, and the role of logic therein; define or recognize simple and compound propositions; use truth tables for compound propositions; form converse, inverse, and contrapositive of a given conditional proposition; use truth tables to determine if a compound proposition is a tautology; use rules of logic, Venn diagrams, or truth trees to test the validity of arguments.

III. Numeration Systems: define a mathematical system; translate members written in bases other than ten into our decimal system and vice-versa; perform arithmetic operations in bases other than ten.

IV. The Rational Numbers: identify prime and composite numbers; prime factor a number; identify the Fundamental Theorem of Arithmetic; find the greatest common factor; find the least common multiple; recognize the tests for divisibility; identify the properties for whole numbers - identity, commutative, associative, closure, distributive; identify

integers; perform the operations of addition, subtraction, multiplication, and division with integers.

V. Mathematical Systems: define a mathematical system; perform arithmetic operations in Clock Arithmetic; define a finite system; define the properties of closure, associative, commutative, etc. for a given operation; define modular arithmetic; define the rules and laws that are obeyed by abstract mathematical systems.

VI. Algebra: solve linear equations with one operation; recognize like terms; solve linear equations by combining like terms and using the distributive property and several inverse operations; solve verbal problems; translate verbal expressions into mathematical expressions; identify a ratio; identify a proportion; solve proportions.

VII. Geometry: define points, lines, and planes; define a broken line, closed, not closed, simple, not simple, polygon, sides of a polygon, vertices of a polygon, concave polygon, convex polygon, interior points of a polygon, exterior of a polygon, regular polygon; use area formulas and apply the Pythagorean Theorem; find the area and the circumference of a circle and also find the radius and diameter; find the volume of various geometric figures.

VIII. Counting Methods: Count by listing; state the Fundamental Counting Principle; define permutations and N factorials; define combinations; compute problems related to specified material.

IV. Probability: define sample space; compute probability; solve problems involving the Fundamental Principle of Counting; compute probabilities; define conditional probabilities; determine whether two events are independent; solve the probability of two independent events; define mathematical expectation.

X. Introduction to Statistics: prepare a frequency distribution; define the three measures of central tendancy; calculate the mean, median, and mode; calculate the range and standard deviation; apply the properties of the normal curve to solve problems; construct statistical graphs.

XI. Computer Literacy - a brief history of computers, the types of computers, the basic capabilities of the computer, the components of the computer, the various languages used in programming, beginning programming in BASIC which includes the commands: HOME, LIST, NEW, RUN. The structure of the various statements: PRINT, GOTO, For/Next, Read/Data, If/Then, Input, REM; and the practical application of the computer in solving problems from other units in the text.

COURSE ORIENTATION FOR SM 112

Entry Requirements

Appropriate score on the Math Placement Test.

Classroom Procedure

The student will participate in lecture discussion with the whole class and the instructor for 4½ hours each week which is structured so that you will be able to complete the course in one term. You will be required to answer practice problems during the class period and for homework. You will be required to participate in support programs that are necessary and beneficial to the student's successful completion of the course.

Materials of Instruction

A. Required text: Mathematical Ideas, Charles D. Miller, Veren E. Heeren; Scott, Foresman and Company, 1982, Fourth Edition.

B. Supplemental Booklist
(Books and periodicals available in RACC LRC)

Martin Gardner's Sixth Book of Mathematical Games from Scientific American	Gardner,	1971
Mathematics: A Modern Introduction	Keedy,	1970
Introduction to Finite Mathematics	Kemeny,	1974
Mathematics Puzzles and Pastimes	Bakst,	1965
Mathematics Diversions	Hunter,	1963
New Math Puzzle Book	Longley,	1970
Games, Gods and Gambling	David,	1962
Concepts of Probability	Mullins,	1972
Introduction to Mathematics	Meserve,	1969
Mathematical Discovery	Polya,	1962
The World Book of Odds	Neft,	1978

Monthly Publications

Mathematics Magazine
Mathematics Teacher
Scientific American

C. Audiovisual materials: To be used by the student voluntarily and/or at the recommendation of the instructor.

Reel Teacher cassette tapes (In RACC LRC on Reserve)

Test Chapter	Tape Number	Name of Tape
1	105	EXPONENTS, FACTORS AND MULTIPLES
1	103	SETS. SIBSETS. AND SET OPERATIONS
1	104	VENN DIAGRAMS OF SETS AND WORD PROBLEMS
3	101	MODERN NUMERATION SYSTEMS: BASES 10, 5, 2
3	102	ANCIENT NUMERATION SYSTEMS
4 & 7	106	PROPERTIES: COMMUTATIVE, ASSOCIATIVE, AND DISTRIBUTIVE
4 & 7	108	PROPERTIES: IDENTIFY ELEMENTS, INVERSE ELEMENTS AND CLOSURE
5	107	SUJECTS OF REAL NUMBERS
4	109	OPERATIONS WITH INTEGERS
10	111	PERMUTATIONS AND COMBINATIONS
11	112	PROBABILITY

Text Chapter	Tape Number	Name of Tape
14	110	SCIENTIFIC NOTATION
12	113	CALCULATING MEASURES OF CENTRAL TENDENCY
12	114	USING MEASURES OF CENTRAL TENDENCY
12	115	THE CHI-SQUARE TEST

Mathematics and the Liberal Arts cassette tapes by Jack C. Gill
(In RACC LRC on Reserve)

Test Chapter	Tape Number Mod-Lesson	Name of Tape
1	Mod II Lesson 1, 2, 3	Basic Elements of Set
2	Mod II Lessons 1, 2, 3	Elements of Logic
3	Mod IV Lesson 1, 2, 3	Decimal Notation and Bases Other Than Ten
11	Mod V Lesson 1, 2, 3	Basic Elements of Probability
4 & 5	Mod V Lesson	Basic Elements of Statistics
	Mod III Lesson 1, 2	Basic Elements of Algebra
	Supplementary Module	A Review of Computational Arithmetic

D. Text Chapter — Disks

Chapter 14

APPLE Presents APPLE. Know your APPLE II E

Chapter 1

Lesson in Algebra - Sets

Chapter 4

Eduware - Fractions - V3
Math V25, V29.

PJ807 - Addition of Fractions - V28
PJ808 - Subtraction of Fractions - V28
PJ809 - Multiplication of Fractions - V28
PJ810 - Division of Fractions - V28
PJ805 - Proportion Problems - V28
PJ806 - Percent Problems - V28

Mathware Fractions

Chapter 8

Algebra I
Algebra II
Algebraic Expressions
Order of Operations
Integers/Equations II
PJ805 - Proportions Problems
PJ806 - Percent Problems
Polynomials Practice - Using Titles:
Discovering Algebra;
Solving Equations and Inequalities:
Discovering Algebra

Text Chapter	Disks
Chapter 9	Plane Facts About Geometry: Formulas, Shapes, and Skills
	Understanding Lines and Triangles Geometry Basic
Chapter 3	Lessons in Algebra - Exponents
Chapter 7	PJ803 - Modular Arithmetic
Chapter 10	PC739 - Permutations I PC740 - Permutations II PC741 - Combinations PC742 - Permutations and Continuations
Chapter 11	PC743 - Probability
Chapter 12	PC747 - Statistics I PC748 - Statistics II

E. Text:

Text Chapter	Video
Chapter 1	Sets and Sets Relations Cartesian Products and One-to-One Correspondence
	Set Operations and Properties
Chapter 2	Compound Propositions and Their Truth Tables
	Converse, Inverse, and Contrapositive
	Venn Diagrams and Valid Arguments
Chapter 10	Fundamental Principles of Counting and Permutations
	Combinations
Chapter 11	Introduction to Probability

E. Filmstrips -

Chapter 11-12
Introduction to Probability and Statistics - Sample probability, permutations and combinations, the binomial distribution, the normal curve and binomial distriution, sampling and the central limit theorem.

Chapter 14 -
How a Computer Solves a Problem.

F. Additional materials are listed in a Math Media Bibliography available in the library.

Attendance

Students are expected to attend class on a regular basis. The student is responsible for all work covered during any class session that was missed, as well as the work assignments, etc. Due to an increase in absenteeism on the days of scheduled exams, only illness or extraordinary circumstances will be a valid reason for a make-up exam.

Students are expected to wait at least 10 minutes beyond the start of a class session for their instructor, unless otherwise instructed.

Course Content

1. Chapter 14 - Use as a reference
2. Chapter 1 - Sets 1.1 to 1.5
3. Chapter 2 - Logic 2.1 to 2.6
4. Chapter 4 - The Rational Numbers 4.1 to 4.4
5. Chapter 8 - Algebra 8.1 to 8.3
6. Chapter 9 - Geometry 9.1 to 9.4
7. Chapter 3 - Numeration Systems 3.1 to 3.6
8. Chapter 7 - Mathematical Systems 7.1 to 7.6
9. Chapter 10 - Counting Methods 10.1 to 10.6
10. Chapter 11 - Probability 11.1 to 11.4
11. Chapter 12 - Statistics 12.1 to 12.4

Grades

There will be chapter exams. Please check the daily schedule for the dates on which the exams will be administered.

The student's grade depends upon the average of all tests administered (100 points each). All turn-in problems and/or quizzes will act collectively as 100 points (carrying the same weight as one test).

90 - 100 = A	70 - 79 = C
80 - 89 = B	60 - 69 = D

0 - 59 = R

Notification

This course outline is provided for use only as a guideline and is in no way a binding contract between student and faculty. The outline may be subject to changes which are necessitated by circumstances arising during the course. All such changes will be given to all students whenever possible.

Note

Students are advised to review the College Catalog with respect to the "R" grade (repeat) and the "W" grade (withdrawal).

Office

Office is located in Room 213; office hours are posted in Room 515 and in Room 215; please see me or call me at Ext. 325 if you are having any problems.

COURSE DESCRIPTION
FOR
SM 101 DEVELOPMENTAL MATH
3 CREDIT HOURS

TEXT: Basic Mathematics for College Students
by
Linda D. Falstein

Ann Flamm, Assistant Professor
Reading Area Community College

This course is a review of arithmetic and algebra. The student will review rational numbers and decimals. The student will be introduced to sets, their notation and operations; algebraic manipulations dealing with whole number integers and rationals; operations on decimal number and percents; ratios and solving proportions; manipulation of whole number exponents; algebraic manipulations dealing with equations and polynomials; and beginning computer literacy.

OVERALL COURSE OBJECTIVES

This review will help you to:

1. review problems in addition, subtraction, multiplication, and division of whole numbers.
2. find primes and lowest common multiples.
3. solve problems containing rational numbers.
4. use decimals and percents in basic operations.
5. solve ratio and proportion problems.
6. solve problems dealing with algebraic manipulations of integers.
7. manipulate whole number exponents.
8. solve problems containing polynomials using basic operations.
9. solve algebraic equations.
10. understand the basics of set theory.
11. solve word problems containing applications of above material.
12. use the computer as a calculator, tutorial mechanism and useful tool.

COURSE ORIENTATION FOR SM 101

Entry Requirements

An appropriate score on the Math Placement Test.

Classroom Procedure

The student will participate in a lecture discussion with the whole class and the instructor for 4½ hours each week which is structured so that you will be able to complete the course in one term. You will be required to answer practice problems during the class period and for homework. You will be required to participate in support programs that are necessary and beneficial to the student's successful completion of the course.

Math Lab

The Math Lab is located in Room 202. The Math Lab is staffed with a qualified instructor to assist the individual student with any problems that may be encountered in the course work. You will be required to attend the Mathematics Lab 1½ hours a week (preferably in ½ hour blocks of time). You may use this time to retake exams, seek individual assistance, or study independently. The hours when the Math Lab will be in operation will be posted at the beginning of each term.

Apple Computer

The Apple Computers are located in the LRC on the second floor and in Room 523. We have disks containing course content material to be used with the Apple Computer. These disks are housed in the library at the circulation desk. You will be required to make an appointment with the instructor to receive an orientation on the computer. You will also be required to use an appropriate disk and review material related to the unit under discussion. You will therefore have to schedule time in the computer lab so this is accomplished for every unit.

Attendance

Attendance is taken during each lecture and lab period so that the instructor can determine the people who are participating but are having difficulty learning the concepts. Since participation in class and lab is important to the completion of all units, you may miss a total of 3 hours or 2 class sessions. After the third hour or third absence, the given grade will drop a letter grade. Any other missed classes or lab periods will result in an automatic R for the course unless discussed with the instructor prior to the absences.

Student Objectives

A. The student will master the contents of each unit by completing related exercises in the text. These exercises are indicated on worksheets provided by the instructor.

B. The student will complete the chapter review upon finishing the above exercises. The student will achieve an 85% level of accuracy on the review. If the student does not achieve this level, he will be assigned exercises until he masters the material and can achieve an 85% on the chapter reivew. If the student does not complete the review, he will not be allowed to take the Chapter Test. After failing to complete three reviews, a given grade will drop a letter grade.

C. The student will achieve an 85% level of accuracy on the comprehensive chapter test upon finishing the exercises and the chapter review. He will not proceed to the next unit until this is accomplished. He may retake the chapter test one time to achieve the 85% level. The exam may be retaken after the student proves to my satisfaction that he understands the material misssed on the first test. The student will be given the highest grade obtained on any one test. If a student misses a test or if a student has to retake a test, this must be accomplished before the next test is given. Tests will be retaken on a day designated by the instructor.

D. The student will take the comprehensive final upon completing objectives A, B, & C to exit the course. The final may not be retaken.

E. As soon as the student achieves an 85% level of accuracy on the comprehensive tests and has taken the final exam, he then has the option to examine material related to the mathematics course in his curriculum.

COURSE CONTENT

Overview of the Computer
Chapter One - Whole Numbers (a quick review)
Chapter Two - Fractions
Chapter Three - Decimals and Percents
Chapter Four - Ratios and Proportions
Chapter Five - Signed Numbers
Chapter Six - Exponents and Scientific Notation
Chapter Seven - Polynomials

GRADES

The final grade will be determined by averaging the tests and the comprehensive final. Attendance, additional assignments, and chapter reviews will also be taken into consideration.

A - 93 or above grade point average and the completion of all units and final exam
B - 89 to 92 grade point average and the completion of all units and final exam
C - 85 to 88 grade point average and the completion of all units and final exam
R - average less than 85

NOTE: Students are advised to review the College Catalog with respect to the "R" grade (repeat) and the "W" grade (withdrawal).

OFFICE

Office is located in Room 213; office hours are posted in Room 515 and in Room 215. Please see me or call me at Ext. 325 if you are having any problems.

NOTIFICATION

This course outline is provided for use only as a guidline and is in no way a binding contract between student and faculty. The outline may be subject to changes which are necessitated by circumstances arising during the course. All such changes will be given to all students whenever possible.

Materials of Instruction

A. Required Text:

Basic Mathematics for College Students, by Linda Falstein; Addison Wesley, 1982.

B. Filmstrips

Chapter 1, 2, 3

Problem Solving in Math
Whole Numbers, Fractions, Decimals, Percents and Combining Problem Solving Skills
QA 63, p. 76

Chapter 1
Stumbling Blocks in Arithmetic - includes regrouping in subtration, the two-place multiplier, and the two-place divisor.
English and Spanish
QA 115

Chapter 2
Stumbling Blocks in Fractions - language of fractions equivalent fractions, addition and subtraction of fractions, and multiplication and divison of fractions
English and Spanish
QA 117

Chapter 5, 6, 7
Why of Simplifying Expressions

Computers
How a Computer Solves a Problem

C. Phonocasettes

Chapter 1
Supplementary Module Audio Tutorial 1
"A Review of Computational Arithmetic"

Chapter 2
Lesson 105
Exponents, Factors and Multiples

Chapter 5
Lesson 109
Operations With Integers

Chapter 6
Lesson 110
Scientific Notation

D.	General Math	Corvus Volume 25, 29, 14, 18,
	General Computers	APPLE Presents APPLE Know Your APPLE II E
	Chapter	Disks
	1	Division skills Corvus V25, V29 PJ802 Multiplication V28 PJ803 Division V28 Order of Operations MECC - Elementary Math a. Learning to Round Off Numbers b. Developing Speed in Math Skills - Add, Subtract, Multiply, Divide
	2	Eduware Fractions - V3 PJ807 Addition of Fractions - V28 PJ808 Subtraction of Fractions - V28 PJ809 Multiplication of Fractions - V28 PJ810 Division of Fractions - V28 Corvus - V14 Mathware - Fractions
	3	Eduware Decimals - V4
	4	Mathware Ratio/Percents PJ805 Proportion Problems - V28
	5	Algebra I Algebra II Order of Operations Integers/Equations II Algebraic Expressions
	6	Algebraic Expressions Lessons in Algebra Exponents
	7	Algebraic Expressions Polynomials Practices Using Tiles: Discovering Algebra

Appendix F:

Computer Literacy Unit

COMPUTERS

I. History

A computer is an electronic device that does arithmetic. The Abacus was made about 2,500 years ago and was actually the first "computer" because it provided a way to add and subtract on a device and get an answer that was stored on the device.

The tables in math books were made by people who were hired to add, subtract, multiply, and divide numbers to come up with the values that were put in the tables. Blaise Pascal, a mathematician in the 1600s, created a machine which had gears and levers that could add and subtract and give the person an answer. This machine was similar to the plastic adding machines used to keep track of grocery bills.

Charles Babbage, a 19th Century English mathematician, started to work on a machine that would do mathematical calculations and carry out complex instructions. The machine was never built in his lifetime, but his design provided us with all the concepts that are in computers today (Input, Output, Memory, Control, and Arithmetic/Logic). His ideas earned him the name "The Father of the Computer".

Around 1890, an American named Hollerith created a tabulating machine which processed punched cards for the Census Bureau. Hollerith later started his own company and after merging with some small companies, one of the most well-known computer companies of today was formed - IBM. From 1930 to 1940 an American, Howard Aikens, developed and built a machine that was called the Mark I. It was financed by IBM and built at Harvard University. It was electric powered but had mechanical relays and switches. It was 45 feet long, 8 feet high, had one million parts and did 2 addition problems a second.

James Presper Eckert built a true computer in 1946 called the ENIAC (Electronic Numeric Integrator and Calculator). It had 19,000 tubes and did 1,000 problems a second. This was the first all-electronic machine.

Around 1950 the problem of rewiring the computer for each problem to be solved was overcome by John Von Neumann. He developed a way to store a series of instructions called <u>programs</u> inside the computer.

The first computers were very large and very expensive to run and maintain. The first users were banks, insurance companies, and government agencies. The first commercial computer which could be programmed without rewiring was the UNIVAC I. The UNIVAC I, which was first offered for sale in 1950, could do just about anything that

today's computers can do. It was sold to the U.S. Census Bureau. The speed at which calculations are done and the size of the machines have changed. The size was dramatically affected by the transistor which Bell Labs developed in 1971. These replaced the vacuum tubes.

Another development which made computers smaller and more affordable was a new invention that allowed any number of transistors to be etched on the same piece of silicon connected by circuits. These small silicon chips are called integrated circuits.

II. Types

There are three types of computers:

1) A micro-computer is a completely self-contained computer. The main memory and all components are in one spot and controlled by the computer.

2) A mini-computer is connected to terminals which communicate with the computer. The computer is usually isolated in a separate room (such as the bank tellers use to access your account).

3) Macro Network, and Mainframe all refer to a system of hooking up mini-computers.

III. Basic capabilities

1) Memory - the ability to store and retrieve information
2) Math computations - the ability to handle complicated algebraic computations
3) Graphics - the ability to represent output pictorially
4) Logical decisions - the ability to decide what to do next based on an answer derived in the computer
5) Sound - the ability to make sounds
6) Color - the ability to output to the screen different colors
7) Speed - the most important capability of the computer

IV. Components

All computer systems consist of five hardware components: 1) Input, 2) Output, 3) Control, 4) Arithmetic/Logic and 5) Memory

Input - any hardware device that is used to put data into a computer is called an input device. Magnetic tapes and cassette tapes store data as electronic impulses. Hard disks and floppy disks also store programs as magnetic impulses the same as magnetic tape. The information is more accessible on a disk. If you wanted to run a specific program on a tape, you would have to fast forward and check to see what program was at the location where you stopped. You would have to fast forward, or rewind and check again until you located the proper spot.. This all takes time. A disk can be compared to a record. If you want a particular song on a record, you would look at the label and set the arm down on the right "grooves" and immediately have your song. In a computer disk the proper information is obtained just as quickly because the disk drive has a way

of finding information by using its directory to find the address. Floppy disks and hard disks are used for short-term storage of information, or storage of information that is accessed or changed very often. Punched cards and punched paper tapes are not used very often because a tiny rip, tear, dent or fold will render the device useless. The computer accepts information from this type of input through readers. The readers "read" the position of the holes and transfer this to "on-off" switching (binary code) that is understood by the computer. Magnetic ink characters are the numbers you see at the bottom of checks. A reader "reads" the numbers by noticing the location of the "blobs" of ink that are positioned on the numbers. The computer then converts this information back into the name of the bank, the name of the person who has the account and the number of the check. U.P.C. (Universal Product Code) consists of the lines you see on grocery store items. Mark sense is the form of input device you use when you take a standardized test like the SAT or DAT. Both of these devices put information into the computer by use of a light sensor and a series of "on-off" switching (binary code) determined by either a white mark or a dark mark. The keyboard is an input device that directly puts information into the computer.

Output is the process of getting information out of the computer. You will notice that most input devices are also output devices. "Read-only" input devices are those in which computers only read the information and do not change it. These input devices cannot be used to output, such as mark sense, UPC, etc. There are a few types of outputs that are not normally used for input. The printer is used to put information out in the form of "hard-copy", or a printout on paper; the CRT displays output on the screen. The robot arm can be made to do work that is controlled by commands from the computer, and the computer can output both music and a voice with the aid of a speech-synthesizer.

The Control section of a computer is located inside the machine. It is this part of the machine that allows an average individual the ability to use a computer. Part of the control section is the "housekeeping" division which handles all the internal workings of the other four parts. The other division is called the compiler. It takes the instructions which we type in a language similar to English and translates them into binary "machine" language which the machine understands. Researchers have developed an "autoprogramming" system for computers. A user is able to instruct the computer by speaking English.

Processing, also known as the Arithmetic/Logic section, is the section which performs all the actual work in the computer. It contains the rules for mathematical functions such as addition, subtraction, multiplication, division, and exponentiation (raising to powers). It also handles all logical decisions using logical operators such as and, or, if...then, else, equal to, greater than, less than, etc.

Memory - storage is divided into two parts: external and internal. The external part of storage is located outside the machine. This can be magnetic tape or disk, punched cards, paper tape, or solid-state devices. Internal storage is also called memory. It consists of two parts: ROM and RAM. ROM (Read Only Memory), which is factory installed, is a permanent type of memory. It does not need continuous flow of electricity to

maintain its memory. However, ROM is not available to be changed or adapted by the user. A user only has the ability to READ the information in the chips. RAM (Random Access Memory known as (temporary) memory, is user accessible. That means that you, as the user, have the ability to fill the RAM memory space with your information or with "canned" programs that you load. It comes from the factory completely blank. Every time the power is interrupted to the computer, the RAM is erased. This means that if you have been programming for several hours and you are just about finished when the power is interrupted everything that was in the RAM will be wiped out and when power is restored it will be totally blank.

Memory is measured in terms of kilobytes (K). One K is equal to 1,024 bytes of memory. A byte which is the equivalent of one letter, number or symbol, is made up of eight bits. A bit is the smallest particle of information that the computer can recognize. A bit is always one of two things: a 1 or a 0, which is converted into on-off switching. Each letter, number, or character has a distinct code in the binary (machine) language spoken by the computer. This is why you must always use the number zero and number one instead of the capital O and the small L. To the computer each of these is different. (Refer to Chapter 3 in your text.)

All computers handle information or programs the same way. An input device puts information into the computer. As it comes in, the control section directs it to the memory where it is stored. You must "load" a program into memory before it can be used. After it is in memory, then it can be run, or used. Once the program is in memory, control takes over and reads through the program. It rearranges and performs operations requested by the program. If numerical operations must be performed, or logical decisions must be made, the control section transfers information to the arithmetic/logic section where the work is done, and control stores the answers back into memory. When control gets to the part of the program where it is told how to input the "answers", control moves the information from memory to the proper output device.

Actually the control and the arithmetic/logic section do all the "work" of the computer. These two pieces together are the microprocessor. A microprocessor and the memory make up the CPU (Central Processing Unit). The CPU is sometimes called the "brain" of the computer, because it controls the working of the computer, finds all the answers and coordinates all the functions of the computer.

You could have an input device in Pennsylvania, a computer in Florida, and the output device in California, and as long as they were connected by modems (through the telephone), they would work as a computer. Any device that is connected to a computer is called a peripheral, and all the "cards" and cables used to connect these peripherals are called interfaces. So, all of our input and output devices are peripherals that are interfaced with the computer.

Languages

Microcomputers can use several different languages; some were designed especially for micros, while others are from mainframe computers. Some are very general languages designed for beginners; others are specialized languages designed for computer professionals.

FORTRAN

In 1954, an IBM-sponsored committee headed by John Backus began work on a scientific-mathematic language. The result of this effort was FORTRAN (FORmula TRANslator) which was introduced in 1957. FORTRAN is a scientifically-oriented language. In the early days, use of the computer was primarily a scientific occupation, associated with engineering, mathematical and scientific research tasks. FORTRAN is noted for its simplicity and brevity. Its primary purpose is the execution of complex formulas in economic analysis and engineering.

BASIC

BASIC, the most popular language among microcomputer users, was developed in 1964. Originally designed for a mainframe computer, it has translated well into microcomputers. From its beginning, BASIC was designed to be interactive in that the user gets an immediate response from the machine when an instruction is typed in. It is a low-level language which means that programs run in BASIC are very slow compared to high-level language. The major advantage of BASIC is that it was designed for a beginning user. It is constucted to be a very logical and structured language. This is also its greatest disadvantage, as it takes many steps in BASIC to program rather simple procedures.

Many versions of BASIC are in use today. In fact, every microcomputer presently on the market has its own version of BASIC. This means that there is (among others) Apple BASIC, Atari BASIC, Commodore BASIC, IBM BASIC, Radio Shack BASIC, Texas Instruments BASIC, Microsoft BASIC, and many others. These BASICs are not directly interchangeable, however, MICROSOFT BASIC (available for many machines) is the closest thing to a universal BASIC that presently exists.

COBOL

COBOL (Common Business-Oriented Language) is one of the oldest computer languages still in use today. Developed specifically for business in 1960, it is a very wordy, easily understood language. Its commands consist of English-like words and phrases that can be read and understood by non-programmers. Its greatest strength lies in generating reports which can be used in business.

Its greatest disadvantage lies in the fact that it is wordy. It also uses a great deal of memory. This is not a problem on mainframe computers, but it is a considerable disadvantage on microcomputers which have far

less memory to spare. This is one problem that is becoming less important as micros have more and more memory squeezed into them.

Pascal

Pascal was developed in 1971 as a language intended to be easy to learn, easy to retain, and very structured. One advantage of Pascal is that completed programs run much more quickly than a comparable program in BASIC. Another advantage is that all variables and their types must be named and explained at the beginning of each section of programming. This allows most Pascal programs to be relatively free of errors and easy to modify.

BEGINNING TO PROGRAM

You are now ready to begin actually working with the computer. Remember the computer only does what you tell it to do. The first thing that we want to do is to try out the keyboard. Take a minute and press every key on the keyboard. Use the shift key to obtain the $,#,",!,%,( ), etc. Don't worry - although you may get some strange results, you will not harm the machine. Note what happens when you press each key-watch for some suprises!! Go ahead and try!

Now let's see what we can do with the mess on your screen. Press the Return key. Now type the word HOME and press the Return key again. The screen should be clear of all writing and there should be a small blinking square of light in the upper lefthand corner. This square, called the cursor, will appear to mark the spot where the next letter typed will appear on the screen.

All computers use these symbols for math:

+ is addition
- is subtraction
* is multiplication
/ is division
↑ is exponentiation (raising to a power) This character is above the N
() used to change normal order of operations

Give the computer an easy problem to do. Make sure you type in exactly what is below.

2 + 2

What is the computer doing? NOTHING?! Until you press the Return key the computer doesn't know that you have asked it to do anything. Now press the Return key. What answer did the computer give you? The cursor appears with nothing following it.

We need to give it more directions. We told it to solve the problem, but we didn't tell it to give us the answer.

WHENEVER WE WANT THE COMPUTER TO SOLVE A PROBLEM, WE MUST TELL IT TO SHOW US THE ANSWER

We do this by using the command PRINT. This tells the computer to print the answer on the screen. Let's try that.

PRINT 2 + 2

Press the Return key. The answer is 4. Now let's try another problem.

PRINT 2839 + 684 (never use a comma to indicate the numerical groupings)

Press the Return key. Let's try a problem involving multiplication. The symbol for multiplication is the asterisk (*). You must use the shift key to obtain the asterisk.

PRINT 5 * 5

Did the computer show the correct answer of 25? OK, everything seems to be working right now. Try to get the answers to these problems:

21 + 54
16 - 8
23 * 8
87 - 5
8/2 (division)
24/8 (division)

Did you remember to type the command PRINT before each problem and to press the Return key after each one?

The computer gives a "short-hand" way to represent the Command PRINT, and that is ? (use the shift key). So to find the answer to 25 + 3 * 3 you would type: ?25 + 3 * 3, press Return, and the computer will come back with the correct answer 34.

For those of you who expected the answer to be 84, remember the computer will use the correct mathematical order of operations:

1. Work in parentheses
   Exponentiation

2. Multiplication and division in order from left to right.

3. Addition and subtraction in order from left to right.

So, in our problem the computer multiplies 3 x 3 then adds 25.

In other words, if we wanted the computer to add 25 and 3 first, then multiply by 3, we would write the problem as:

?(25 + 3) * 3

Then the computer would do what was in parentheses and the answer would be 84.

To do math problems on the computer, it is necessary to write the problems in BASIC first, then type it on the computer and Return it to get the answer.

Example:	Problem	Answer
	8 + 5 - 6	7
	25 / 5 x 3	15
	$\frac{6 + 8 + 10}{3}$	8

Try the following problems:

1. 6^2
2. (11 + 4) / 3
3. 2^3
4. (19 + 5) / (5 + 3)
5. (16 - 2) / (2 x 2)

Now let's try this one:

```
PRINT "2 + 2"
```

Did the computer give you an answer? No, it printed exactly what you asked it to print!

WE USE QUOTATION MARKS (" ") TO TELL THE COMPUTER TO PRINT EXACTLY WHAT WE HAVE PRINTED. IT WILL NOT SOLVE ANY PROBLEMS BETWEEN THESE MARKS.

Let's try to get the computer to both print and solve a problem in one line.

```
PRINT "2+2="; 2+2
```

Is the output on the screen 2 + 2 = 4? Why? We told the computer to print the 2 + 2 exactly as we typed it and then solve the problem 2+2. Try some more:

```
?   "32 + 10 ="; 32 + 10
?   "164 - 121 ="; 164 - 121
?   "83 * 4 ="; 83 * 4
```

If you want to clear the screen, type HOME. The cursor should be in the upper left corner.

Since the computer seems to be handling numbers pretty well, let's try to use it for words. Try this one:

```
PRINT HELLO THERE PERSON
```

What happened? It didn't print the words we typed. WHY NOT? The answer lies in the origins of the BASIC Language. It was invented by mathematics professors for use by their students. The main function they wanted to have was an easy use of numbers. It is a bit more difficult to use words. Try this:

```
PRINT "HELLO THERE PERSON"
```

Does the output on the screen say HELLO THERE PERSON? What was the difference? Of course, it was the use of the quotation marks. The quotation marks are a signal to the computer to print exactly what we have typed in. Let's try a few more examples:

```
PRINT "I'M YOUR COMPUTER"
PRINT "I LIKE YOU"
PRINT "YOUR NAME"
PRINT "I'M ENJOYING THIS"
```

FIRST PROGRAM

Until now you have been working in the IMMEDIATE MODE. As soon as you pressed the RETURN key, the computer executed your request. You are now ready to begin writing your first program. Programming is done in the DEFERRED MODE. This means that when you press the Return key, the computer will store your instructions until you ask it to execute them.

First, we must prepare the computer by making sure there are no instructions in its memory. We do this by using the command NEW. Type in the word NEW and press the RETURN key. The NEW command clears the RAM of all previous instructions. Type HOME. Press Return. At this point you should see a blinking cursor. If you don't have this, check to make sure you have followed the Power-Up procedures properly.

All programs have certain features in common. One of these features is that a program is made up of lines, each having a separate LINE NUMBER. Let's try writing a program!

```
10 PRINT "HELLO THERE"
```

Press the RETURN key after you have typed this line. Why didn't the computer do anything? Because the computer recognizes that this is a PROGRAM. How did the computer do this? When you began your line with a number, it sent a signal to the COMPILER portion of memory that this was a LINE NUMBER from a program. Now the computer is waiting for further instructions. Let's tell it to execute or program. The command word to begin execution is RUN. Type in RUN (no line number) and press the RETURN key. Did the words HELLO THERE appear on your screen? What happened to the line number, PRINT, and the quotes? They are instructions to the computer and are never shown when a program is run.

Try to RUN the program again. Type the command RUN. Press Return. Try it again.

ONCE A PROGRAM IS ENTERED INTO THE COMPUTER IT STAYS THERE UNTIL THE POWER IS SHUT OFF OR IT IS REMOVED BY TYPING "NEW".

Let's add another line to this program. Type in this line carefully.

```
20 PRINT "I'M YOUR COMPUTER"
```

Press the RETURN key. Now RUN this program. Does the output on your screen look like this:

```
HELLO THERE
I'M YOUR COMPUTER
```

If it doesn't, check your program carefully for typing errors. The computer began at line 10, completed it, and then did line 20.

A PROGRAM WILL FOLLOW THE LINE NUMBERS IN STRICT ASCENDING ORDER (STARTING WITH THE LOWEST AND ENDING WITH THE HIGHEST)

At this point your screen has many sentences on it. Let's clear away the clutter. Type HOME. What happened to everything? Did we lose our program? To check, try to RUN the program we had. Type RUN and press RETURN. The program still ran which means the computer still remembers it. But we cleared everything from the screen.

HOME ONLY CLEARS THE SCREEN. IT DOES NOT CLEAR OUT THE MEMORY THE WAY NEW DOES.

We can use the word HOME to clean the screen off anytime that it gets too cluttered, without losing our program.

Now let's try to change the program. Type in this line:

```
10 PRINT "HI (YOUR NAME)"
```

Now run your program. Was what appeared on the screem what you expected? What happened to "HI (YOUR NAME)"?

THE COMPUTER ACCEPTS THE LAST LINE TYPED IN AS THE ONLY CORRECT LINE WITH THAT LINE NUMBER.

This provides us an easy way to change our mistakes. We only have to retype a line in order to correct a mistake. As far as the computer is concerned, it only remembers the last time we input a line number.

Let's put a few more lines in.

```
30 PRINT "HERE IS A MATH PROBLEM"
40 PRINT "1 + 1 = "1 + 1
50 PRINT "END"
```

Now RUN this program. Try to add a few more lines on your own.

We can use the print command or ? to change the output of the screen

display. One way to do this is to use a PRINT blank statement. Type this example:

```
10 PRINT "YOUR NAME"
20 "NOTHING PRINTS FROM THE FOLLOWING LINE"
30 PRINT
40 PRINT "SEE WHAT I MEAN"
```

You told the computer to print a blank line in Line 30 so it did. We can use a blank PRINT statement anytime that we want to have a space printed on the screen.

You can delete a line by typing the line number and pressing return.

Try this program:

```
10 PRINT "RUN THIS PROGRAM"
20 PRINT "TO SEE WHAT"
30 PRINT "OMIT THIS LINE"
40 PRINT "HAPPENS"
RUN
```

Type 30 and press RETURN. Then type RUN. Check to see that Line 30 was omitted.

REM/HOME/LIST

When you write a program, many times it is helpful to include descriptions of what the program is designed for, what each part does, and when you did it. We can include that kind of information within the listing of the program by using REM statements. REM, which is short for Remark, is a line that the computer ignores when the program is running, but which contains information for the programmer. The only time that a REM appears is when you List a program. Try this sample program. Be sure to NEW your computer first.

```
10 REM THIS IS A COMMENT
20 REM PRINT "THIS WON'T PRINT"
30 PRINT "SEE!  YOU DIDN'T BELIEVE ME"
40 REM PRINT "THIS DOESN'T PRINT EITHER"
```

Try RUNning this program. What does your screen output look like? Why didn't Lines 10, 20, 30 and 40 print? The reason is because of the REM command. When the computer runs, all the REM Lines are ignored. The only time they show up is when the programmer wants to LIST them.

LIST is the command we use to tell the computer to show on the screen everything in its RAM. Enter the word LIST. Press RETURN. The computer puts on the screen the program you just ran. It is the only thing in the RAM at this time. Use the LIST command whenever you wish to look at your program. Notice 10, 20, 30 and 40 are still there, even though they didn't show up when the program was run.

If you want to look at certain lines, type LIST and the specific line numbers you want to see. Type LIST 40, 50 and press RETURN.

The HOME command is another useful programming tool. The screen clears, but everything stays in the memory. We use HOME to clear the screen at various times in the program. Type NEW and press RETURN before entering the following program.

```
10 REM HOME DEMO
HOME
30 PRINT "HERE IS THE HOME DEMO"
40 PRINT "WATCH IT CLOSELY"
50 PRINT "IT'S VERY QUICK"
60 PRINT "HERE IT GOES"
70 HOME
80 PRINT "DID YOU SEE IT?"
```

Run the program. You may have to run it several times to really see it. The computer printed Lines 30, 40, 50, 60, then cleared the screen. Nice, but quick!

PRINT/GOTO

Be sure to type NEW before we begin this section.

One of the most powerful commands we can use is called the GOTO command. It forms an unconditional branching of the program. Type in this program. Instead of the words "YOUR NAME", actually type in your name.

```
10 PRINT "YOUR NAME"
20 GOTO 10
```

Now RUN this program. How long will this thing run? Until the power is shut off or until you press CONTROL/RESET. (Do this now). Let's examine this program to see what happened. First the computer did Line 10 and printed your name. Then, it went to Line 20. Line 20 told the computer to go to Line 10, so it did. Line 10 said to print your name and then proceed to Line 20 which said to go to Line 10 and . . . Get the idea!

Type NEW. Press RETURN. Let's make a change in Line 10.

```
                    10 PRINT "YOUR NAME", (add the comma)
Example:            10 PRINT "ANN", "ANN", "ANN"
```

Now RUN the program. Does it look any different? Is your name printed in 3 print zones across the screen? What was different about this Line 10? The only difference is that you typed a comma after your name.

A COMMA FOLLOWING A "PRINT" STATEMENT TELLS THE COMPUTER THE NEXT "PRINT" STATEMENT IT RUNS INTO SHOULD BE PRINTED ON THE SAME LINE BUT IN THE NEXT PRINT ZONE TO THE RIGHT.

The APPLE screen is divided into 3 print zones. The first print zone is 16 key-strokes, the second print zone is 16 key-strokes and the third print zone is 8 key-strokes. If a comma is found after printing in the third print zone, the computer will print the next "PRINT" statement on the next

line in the first print zone.

```
10 PRINT "YOUR NAME";  (add the semicolon)
20 GOTO 10
```

RUN the program. Now your name fills the screen. Press CONTROL and RESET to stop the program. The semicolon causes the computer to print each item on the same line, one directly following the other. Notice that there are no spaces between the printings of your name.

VARIABLES/LET

Type NEW in your machine.

We use variable names to identify specific places in the RAM of the computer. We can use any of the legal variable names. The names generally can be one or two characters long, but they must begin with a letter. EX: A,B,C,...Z; A1, A2,...Z9.

The command LET is used to name memory locations in the RAM. The form we use looks like this: LET A=1. What this does in the computer is very similiar to what the Post Office does when someone moves into town. When someone moves into a town the Post Office sets aside a mail box for them and puts their name on it to identify what person it belongs to. When mail arrives addressed to that person, they find the mail box with the proper name and put the mail in it.

When the computer sees the statement LET A=1, it sets aside a memory location (mailbox) and labels it as A(Name). When the computer sees the "=" sign (mail arriving) it puts the value that follows the equal sign (mail) into the location (mailbox).

YOU CAN ONLY USE ONE VARIABLE NAME ON THE LEFT SIDE OF THE "=" SIGN.

Try this program:

```
10 LET A = 1
20 LET B = 5
30 LET A1 = A + B
40 PRINT A
50 PRINT B
60 PRINT A1
```

RUN this program. The numbers 1, 5, and 6 are the values that are found in memory locations A, B, and A1.

One interesting feature of using variables is that each time you state a new value for a variable the old one is replaced. For example, if you said LET A = 1 and then said LET A=5, when you printed out the value of A it would be 5. NEW your computer and try this program:

```
10 LET A = 1
20 PRINT A
```

```
30 LET A = 5
40 PRINT A
```

When you run this program, in the first PRINT statment the value of A is 1. The next LET command replaces the value of 1 with the value of 5.

Type in a different Line 30

```
10 LET A = 1
20 PRINT A
30 LET A = A + 1
40 PRINT A
50 GOTO 30
```

Now RUN the program. Press the Control and Reset key and the program will stop running and you can examine the values presented in the column. Each number is one larger than the previous number.

Line 30 tells the computer to take the present value in A, add 1 to it, and make that the value of A.

Type NEW and press Return. Try this program.

```
10 A = 6
20 B = 23
30 PRINT A + B
```

The word LET is optional in most versions of BASIC.

INPUT

In many programs we want the person using the computer (the user), to give us some information we can use with the program. This might include such things as name, dates, numbers and others. The command we use for this is INPUT.

THE INPUT STATEMENT MAKES THE PROGRAM STOP AND WAIT FOR INFORMATION FROM THE USER.

Let's try an example:

```
10 REM INPUT
20 PRINT "TYPE IN A NUMBER AND THEN PRESS ENTER"
30 INPUT N
40 PRINT N
50 GOTO 10
```

RUN this program. Notice that after the sentence TYPE IN A NUMBER AND THEN PRESS ENTER, nothing happens except for the appearance of a ? and the cursor. The computer is waiting for you to give it some information. Type in a number and hit the enter key. The computer has printed your number and asked for another number. This will continue until you press the Control and Reset keys. Let's try using this function. NEW your

machine and type in this program:

```
10 HOME
20 REM MULTIPLICATION HELPER
30 PRINT "THIS PROGRAM FIGURES OUT 5 TIMES
       ANY NUMBER"
40 PRINT "GIVE ME A NUMBER"
50 INPUT N
60 LET A=5*N
70 PRINT "5 TIMES N IS"; A
80 GOTO 40
```

RUN this program. Every time you give the computer a number it returns with an answer that is 5 times bigger. Press the Control and Reset key when you want to stop.

We can also program the computer to answer any two-number multiplication problem. Type NEW, then put in this program.

```
10 HOME
20 REM MULTIPLICATION SOLVER
30 PRINT "ENTER A NUMBER"
40 INPUT A
50 PRINTER "ENTER ANOTHER NUMBER"
60 INPUT R
70 PRINT "A X R = "; A * R
80 GOTO 30
```

RUN this program. (CAUTION - IF YOU USE VERY LARGE NUMBERS THE ANSWERS WILL NOT LOOK LIKE YOU EXPECT, BUT THEY ARE STILL CORRECT.) Press Control/Reset to end the program.

STRINGS ($)

Until this point, every time we wanted output to have words, the command PRINT was used. The reason for this lies in the creation of BASIC. Originally designed by two math professors for use by their students, using words was not high on their list. However, provisions were made to use words with a structure called STRING VARIABLES.

STRING VARIABLES are a series of alphanumeric (numbers and letters) characters considered as a single unit by the computer. In plain English, what this means is that the computer will consider any STRING that it is given as a single word. The STRING can consist of letters, numbers, spaces, symbols, or a combination of these. However, even when the STRING consists purely of numbers, the computer considers it a word and will not add, subtract multiply, or divide with it.

STRING VARIABLES are named for identification in much the same way as numeric variables. The only difference is that STRING VARIABLES are tagged with a $ symbol (SHIFT, 4). Some examples of STRINGS are A$, B$, Y$. Notice that they are the same names as numeric variables except for the $ at the end. Let's try two short programs with STRINGS.

```
10 HOME
20 PRINT "TYPE YOUR NAME"
30 INPUT N$
40 PRINT "HELLO,"; N$
```

Once a string is entered into memory, you can recall and use it as often as you want. Type NEW and enter the next program.

```
10 HOME
20 REM STRING PROGRAM
30 PRINT "TYPE IN YOUR NAME";
40 INPUT N$
50 PRINT "TYPE IN A FRIEND'S NAME";
60 PRINT F$
70 PRINT "TYPE IN A PLACE";
80 INPUT P$
90 PRINT "TYPE IN A TEACHER'S NAME"
100 INPUT T$
110 PRINT "CONGRATUATIONS";N$
120 PRINT "YOU HAVE JUST WON ONE MILLION
        DOLLARS IN OUR LOTTERY!!"
130 PRINT "TOGETHER YOU AND ";F$;
140 PRINT "CAN TAKE THE NEXT PLANE TO ";P$
150 PRINT "ALL YOU NEED TO CLAIN YOUR PRIZE
        IS A NOTE FROM"; T$
160 PRINT "PROVING THAT YOU ARE A STRAIGHT
        'A' STUDENT!!"
```

IF/THEN

The IF/THEN command lets us give the computer the ability to make "intelligent" choices. It is always used within the same statement. The IF/THEN command sets up a test/action for any variable. It works this way: the IF part of the test section is placed on the first part of the statement. If the test set up in the IF section is TRUE the action proceeds to the second section of the statement (THEN) which directs an action. If the first section is FALSE, the program drops immediately to the next line without any other action being taken. This enables a programmer (you!) to decide what answers to questions will be accepted and what action is taken. It also allows you to set limits on the number of times something is to be done. Try this program:

```
10 HOME
20 REM IF/THEN
30 LET A = 2 + 2
40 IF A=4 THEN GOTO 70
50 PRINT "A DOES NOT = 4"
60 END
70 PRINT "A = 4"
```

RUN the program. What is the output on the screen? A=4. Let's see why by tracing the program. Line 10 clears the screen. Line 20 does not execute. Line 30 sets a value in variable A of 4. Line 40 looks at

the value in A and tests to see if it equals 4. Since it does, the first section test is "true" and the action proceeds to the THEN section. The THEN section directs the program to Line 70, so it does Line 70. Line 60 is never done because it is skipped over.
Change Line **30** to read 30 LET A=5. RUN the program again. Were the results the same? No, because the first section of the IF/THEN test was false. The action dropped immediately to Line 50 then to Line 60 where the program ENDed.

NEW your computer and try this program:

```
20 REM IF/THEN
30 LET A = 0
40 PRINT A
50 IF A = 4 THEN GOTO 80
60 LET A = A + 1
70 GOTO 40
80 PRINT "DONE"
```

RUN the program. Line 50 tests the value of A each time through the loop to see if it ='s 4. If A doesn't equal 4 the program drops immediately to Line 60 which increases the value of A by 1. Line 70 directs the program back to Line 40 which PRINTs the new value of A. When the value of A reaches 4 the test is "true", action proceeds along the line to the THEN section which jumps the program to Line 80. Line 80 PRINTs DONE and ends the program.

Change Line 60 to read:

```
60 LET A = A + 3
```

RUN the program. The value of A never equals 4, therefore, the test section (if) is never "true". Press Control/Reset to stop the program. Change Line 50 to stop the program when the value of A becomes greater than 4.

```
50 IF A > 4 then GOTO 80
```

A = A+3 is not valid in math but is allowable in computer talk because = may not mean equal to the computer. It means take the current value of A, add 3 to it and that answer is now the new value of A.

READ/DATA

To this point, you have learned two ways to give values to the program: 1) INPUT statements; and 2) LET statements. There is another way to give information to the program called READ/DATA statements. Like the IF/THEN statements, the READ/DATA statements are always used in matched sets. The biggest differences are that one READ statement can be matched to many DATA statements, and the DATA statements can be located anywhere in the program.

Let's examine exactly what the READ/DATA statements do in a program. The command READ to the computer means very much what you would expect; it

tells the computer to look for a program line beginning with the word command READ to the computer means very much what you would expect; it DATA and, literally, read the first item in the statement. At that point, the computer uses a "pointer" which keeps track of which piece of DATA it has READ, much the same way you would use a bookmark to keep track of what you had read in a book. The next time the computer is instructed to READ, it goes to the next item in the line.

The DATA command sends a signal to the computer that it is a set of information. Whenever you see a READ, look for it. Wherever a DATA statement is located in the program, the computer will always find it when it receives a READ command. It doesn't matter if the DATA statement is located before the READ, or in the middle of a program, or at the very end of the program; the computer will always search until it finds the DATA. The comma used in a DATA line indicates the end of each piece of data. This is why you may never use the comma as a placeholder as in 1,000 for the number one thousand. The computer would read it as 1 and 000. Let's try some examples:

```
10 HOME
20 REM READ/DATA
30 READ X
40 PRINT X
60 DATA 3,5,7,9
70 END
```

RUN this program. What does your screen output look like? Do you have the numbers 3, 5, 7, 9 printed down the left side of the screen? You do not? Line 10 clears the screen, Line 20 is a program remark for information. Line 30 tells the computer to look for a line marked DATA and READ the first item, putting that value in memory location X (remember variables and memory locations?). Line 40 prints PRINTs the value of X. Line 60 is the DATA we're looking for, and Line 70 ENDs the program. Why did only the first value in the DATA line get PRINTed? Because that's all we told the program to do! Line 30 only tells the computer READ once. In order to READ the remaining items on the DATA line, the program needs to be directed back after PRINTing to READ the next item. How can we give it this direction? Hint: Notice the program skips from Line 40 to Line 60; think a Line 50 might be:

```
50 PRINT GOTO 30
```

Notice the error message - ? out of data error in 30. The message tells you that the computer has tried to READ but has found no more DATA items to READ. One way to avoid the error message is to identify the last piece of data. The DATA we supply is called a FLAG. FLAG is data not to be processed. An example of FLAG in a program using prices would be -99. Using the IF/THEN command we set up a test to see if the FLAG is read, and if the FLAG is read, it jumps to another line. Insert line 35 IF X=-99 THEN GOTO 70. Change Line 60 to DATA 3,5,7,9,-99. Now, run the program. Remember that the order of your program should be READ, TEST, PRINT. This will prevent the FLAG from being printed or used.

The computer will find the DATA line wherever it is in the program. Even

when the DATA line comes after the END statement, the computer still searches and finds it. When there is more than one DATA statement, the computer reads the DATA statement with the lowest line number.

FOR/NEXT LOOPS

One of the more useful features that is built into the compiler is that of the FOR/NEXT counter. The compiler has the ability to set up a "counter" with the limits to be set by the user. It then completes an action the specified number of times. NEW your computer and try this program:

```
HOME
20 FOR A=1 TO 10
30 PRINT "YOUR NAME"
40 NEXT A
50 PRINT "COUNT THE NUMBER OF TIMES YOUR
         NAME IS PRINTED"
```

RUN the program. How many times did your name print? Let's trace the program. Line 20 sets up the "counter" for the loop with a bottom limit of 1 and a top limit of 10. Line 30 PRINTs your name. Line 40 advances the count by 1 and jumps the program back to Line 20. When the value of A equals 10, the computer completes the loop and drops to the line after the NEXT statement. Add these lines to the program:

```
60 FOR B = 1 TO 10
70 PRINT "PARTNER'S NAME"
80 NEXT B
```

RUN the program and watch the output. It should print your name 10 times and then print your partner's name 10 times. Change the program so that it prints in all 3 columns.

NEW your machine. Type in this program. It will show you the value that the counter in the compiler has each time through the loop.

```
HOME
20 FOR J = 1 TO 12
30 PRINT "J=";J
40 NEXT J
```

Notice the output. Each time through the loop the value of J increases by one. You can input one or both of the values in the FOR/NEXT statement.

```
10 PRINT "PICK A NUMBER GREATER THAN 1"
20 INPUT N
30 FOR Z = 1 TO N
40 PRINT "THIS IS EASY!!"
50 NEXT Z
60 GOTO 20
```

You can run a FOR/NEXT loop from any number to any other number.

```
10 FOR J = 10 TO 25
20 PRINT "J =";J
30 NEXT J
```

ERIC Clearinghouse for Junior Colleges
8118 Math-Sciences Building
University of California
Los Angeles, California 90024

JAN 4 1985